Are You Contributing Trustworthy Data? The Case for a Reputation System in Participatory Sensing

Kuan Lun Huang[†‡]
klh@cse.unsw.edu.au

Salil S. Kanhere[‡]
salilk@cse.unsw.edu.au

Wen Hu[†]
Wen.Hu@csiro.au

[‡]School of Computer Science Engineering, University of New South Wales, Sydney, Australia

[†]CSIRO ICT Centre, Brisbane, QLD, Australia

ABSTRACT

Participatory sensing is a revolutionary new paradigm in which volunteers collect and share information from their local environment using mobile phones. The inherent openness of this platform makes it easy to contribute corrupted data. This paper proposes a novel reputation system that employs the Gompertz function for computing device reputation score as a reflection of the trustworthiness of the contributed data. We implement this system in the context of a participatory noise monitoring application and conduct extensive real-world experiments using Apple iPhones. Experimental results demonstrate that our scheme achieves three-fold improvement in comparison with the state-of-the-art Beta reputation scheme.

Categories and Subject Descriptors

C.m [**Computer Systems Organization**]: Miscellaneous

General Terms

Design, Performance, Experimentation

Keywords

Mobile Computing, Participatory Sensing, Urban Sensing, Reputation System, Trust, Data Quality

1. INTRODUCTION

The recent wave of sensor-rich, Internet-enabled, smart mobile devices such as Apple iPhone has opened the door for a novel sensing paradigm, *participatory sensing* [1], for monitoring the urban landscape. In participatory sensing, ordinary citizens collect data from their surrounding environment using their mobile devices and upload them to an application server using existing communication infrastructure (e.g., 3G service or WiFi access points). The application server then combines data from multiple participants, extracts the community statistics, and uses them to build a spatial and temporal view of the phenomenon of interest. Several exciting participatory sensing applications have emerged in recent years. Cartel [2] is a system that uses mobile sensors mounted on vehicles to collect information about traffic, quality of en-route Wi-Fi access points, and potholes on the road. This revolutionary paradigm is also being used to collect and share data about air pollution [3], noise pollution [4, 5], cyclist experiences [6], diet [7] and pricing information of consumer goods [8, 9].

The success of the above applications requires a high level of participation from voluntary users. Unfortunately, the very openness which allows anyone to contribute data, also exposes the applications to erroneous and malicious contributions. For instance, users may inadvertently position their devices such that incorrect measurements are recorded, e.g., storing the phone in a bag while being tasked to acquire urban noise information. Malicious users may deliberately pollute sensor data for their own benefits, e.g., a leasing agent may intentionally contribute fabricated low noise readings to promote the properties in a particular suburb. Without confidence in the contributions uploaded by volunteers, the resulting summary statistics will be of little use to the user community. Thus, it is imperative that the application server can evaluate the trustworthiness of contributing devices so that corrupted/malicious contributions are identified. The results can be used to aid further analysis and ultimately provide more reliable outcomes. For example, the server may lower the weights of corrupted/malicious data in the computation of community summary statistics (e.g., average noise level in a neighborhood) so as to acquire a more accurate representation of the phenomenon of interest.

In this work, we propose a reputation system for evaluating the trustworthiness of volunteer contributions in participatory sensing applications. Our reputation system allows the server to associate a *reputation score* with each contributing device that reflects the level of trust perceived by the application server about the data uploaded by that device over a period of time. A high reputation score is an indication that a particular device has been reporting reliable measurements in the past. Hence, it warrants that the server places a higher level of trust in the sensor readings from that device in the future. In [10], Ganeriwal *et. al.* proposed a reputation framework referred to as RFSN, to counter faulty and misbehaving nodes in traditional embedded wireless sensor networks. RFSN is made up of two main components - (i) watchdog module and (ii) reputation module. The watchdog module implements an outlier detection algorithm to detect non-cooperating nodes at each time in-

MSWiM'10, October 17–21, 2010, Bodrum, Turkey.

stant. The resulting node ratings act as input to the reputation module that builds a long-term view about the quality of the contributions from the nodes. In this paper, we adopt a similar architecture as that of RFSN, but propose to use different algorithms to implement the system building blocks that are particularly suited to the unique characteristics of participatory sensing.

We make the following specific contributions:

- We argue for the need of a reputation system in participatory sensing applications to assess the trustworthiness of user contributed data. We propose a reputation system that uses the Gompertz function for rating the contributions made by participating devices. We show that our system is well-suited to quickly adapt to the transitions (e.g., from cooperative to non-cooperative) in user behavior. Such dynamism is fairly typical in participatory sensing. Moreover, our system can be readily incorporated in a variety of participatory sensing applications.

- We implement our reputation scheme within a real-world participatory sensing application for monitoring noise pollution in urban environment. We conduct extensive experiments using Apple iPhones in different scenarios that capture situations in which users contribute corrupted data, both inadvertently and due to malicious designs. The results show that our reputation system outperforms the state-of-the-art Beta reputation scheme by a factor of 3.

The rest of this paper is organized as follows. Section 2 presents an example to motivate the need for a reputation system in the context of participatory sensing. Related work is summarized in Section 3. Section 4 presents an overview of the system architecture. Sections 5 and 6 provide details of the watchdog and reputation modules, respectively. In Section 7, we describe the experimental setup and present evaluation results. Section 8 concludes the paper.

2. MOTIVATING EXAMPLE

In this section, we use an illustrative example from a real-world participatory sensing application to motivate the need for using reputation in such systems. We consider a noise mapping application[1] similar to [4, 5], which generates a collective noise map by aggregating measurements collected from the mobile phones of volunteers. We conducted an experiment using 6 mobile phones, instrumented with a sound level meter (SLM) program. The SLM measures the ambient noise level (when the phone is not used for conversation) and reports the A-weighted equivalent continuous sound level, LA_{eq} (measured in dbA) every second. The experiment was conducted in a typical office environment of size $30m$ by $20m$ by placing the phones on different desks for a duration of 30 minutes. The samples were relayed to a central server over WiFi, which then computed the average value of the ambient noise in the office room from the reported measurements. Further details about the software and hardware used for this experiment are provided in Section 7. To demonstrate the need for reputation, we created a scenario where the devices are operated such that we can capture typical use cases in which some devices contribute corrupted data. In particular, the following placement configuration was adopted. Devices 1 and 2 were kept on the desk with the phone microphone unobstructed. This represents the normal behavior in which users collect data as expected. Devices 3 and 4 were toggled between the following two positions - (i) on the desk, i.e., normal and (ii) inside the drawer. We expect to see a lower LA_{eq} recorded by the devices when placed in the drawer, since the wooden exterior affects the propagation of sound waves. This behavior reflects a plausible scenario, where a participant may sometimes inadvertently position his device in a way that hinders the data collection process (for example, by placing the phone in the pocket or bag). Finally, devices 5 and 6 reflect the behavior of malicious users. Both these devices were placed inside the drawer for the entire duration of the experiment, thus simulating users who intentionally contribute corrupted data. Further, we assume that the user of device 6 is a sophisticated attacker, who has modified the phone software such that a random Gaussian offset is added to the LA_{eq} value computed by the SLM program. Fig. 1 plots

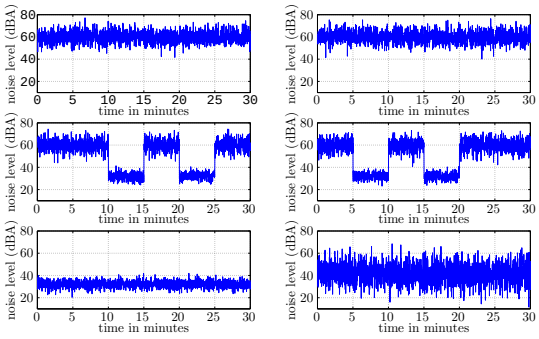

Figure 1: Noise samples recorded by each device, from top left to bottom right (left to right orientation): device 1 to 6

the noise level measured by all 6 devices. These graphs very clearly match the aforementioned behavior of each device. Device 5 persistently reports low LA_{eq} values while devices 3 and 4 show a distinct pattern of high values followed by low values depending on whether they are placed on the desk or in the drawer. Device 6 reports random values due to the addition of the random offset.

Recall that, the objective of the application is to determine the noise level in the office using data from these 6 devices. It is obvious that if the application server resorts to simple averaging, the final result would be erroneous, since this would also include inaccurate sensor readings (for example, data contributed by devices 5 and 6 and devices 3 and 4 when they were placed in the drawer). A better approach would be to associate a weight with each sensor reading, such that the weight reflects the quality of the data, and then computing a weighted average. However, the server has no knowledge of the ground truth (e.g., the server does not know that device 6 is malicious) and hence has to resort to some form of approximation to assign the weights. A common approach is to use consensus-based outlier detection. The weights can be determined by executing a consensus-based outlier detection algorithm, where a group consensus is calculated from the values reported by all devices within one epoch of time. Now, each device is associated with a weight, which is inversely proportional to the deviation between the device sample and the group consensus (e.g., a de-

[1]Even though the above discussion focuses on noise monitoring, the arguments we make here apply universally to other participatory sensing applications

vice which reports a value that is significantly different from the group consensus is assigned a low weight). One problem with purely relying on outlier detection is that it treats each epoch independently. Thus, it is not possible to gain any insights into the behavior of the devices over a long time period, which is valuable in reinforcing the server's confidence about the trustworthiness of the contributing devices. Consider the aforementioned example again and assume an epoch of 1 minute. As can be seen from Fig. 1, during the 5^{th} epoch, the server can readily determine that device 5 is contributing bad data, since the samples reported by this device are significantly different from the common consensus (which is closer to the LA_{eq} collected by devices 1 to 4). Hence, the server knows that measurements from device 5 during this particular epoch should be assigned lower weight. However, it does not have sufficient information to justify the choice of the actual value (e.g., should it be given a weight of 0.1, 0.2 or 0.3, assuming any value less than 0.5 is an indication of corrupted data). Instead, if the server was able to look into the past and observe the behavior of device 5 from the first epoch, then it would become possible for the server to make a more well-informed decision about the weight associated with device 5. For example, if this device had been consistently contributing corrupted data (as is the case in Fig. 1), then its weight should be very low (as a result of gradual weight reduction over the past epochs). Consequently, the application server would be able to arrive at a more accurate estimate of the noise level in the office. In light of the above arguments, we are thus motivated to introduce device reputation as a measure of the trustworthiness of the past contributions of individual devices in the context of participatory sensing applications.

3. RELATED WORK

Reputation systems have long been studied in a diverse range of disciplines. We are all familiar with the way online markets such as eBay [11] and Amazon [12] use reputations to enhance the buying and selling experiences. For example, eBay uses a simple feedback mechanism, where the buyer assigns either a positive, negative or neutral rating to the seller based on his/her satisfaction with the transaction. A member's overall feedback score is simply the difference between the number of unique positive feedback reports and negative feedback reports received in the past 12 months.While this approach is simple to implement and understand, it has some flaws. First, the negative ratings can be easily masked if there exists a proportionately large pool of positive ratings. Further, this scheme has a significant lag (12 months) and hence it may take a long time for the feedback score to reflect a drastic change in the users' behavior (e.g., a shift from genuine to malicious). As such, this simple approach is not viable in our context.

Reputation systems have also been widely used in ad-hoc wireless networks [13, 14, 15, 16]. In [13, 14], the authors borrow the ideas from game theory and attempt to address the selfish routing problem in such networks. In [15, 16], Bayesian analysis is used to formulate a similar problem and the resulting reputation systems are shown to counter any misbehaving nodes. Bayesian reputation systems are quite flexible and can be adapted with relative ease in different types of applications and environments [17, 18]. For example, the reputation framework, RFSN, proposed in [10] makes use of Beta reputation [17] for associating a reputation score with each sensor node in a traditional embedded wireless sensor network. Beta reputation has simple updating rules as well as facilitates easy integration of ageing. However, as we show in Section 7.2, it takes a less aggressive approach in penalizing users that contribute corrupted data. Note that, in participatory sensing applications, the period over which a user may contribute corrupted data may potentially be short-lived (particularly, when this happens unintentionally). Further, the frequency of occurrence of such events may also be high (e.g., a user may frequently place her phone in the bag while it is collecting noise samples). Hence, it is desirable that the reputation scheme is responsive enough to capture such dynamic behavior. In this paper, we propose to use the Gompertz function, which is particularly well-suited to deal with the aforementioned scenarios. Beta distribution has also been employed in [19], where the authors address the problem of selecting suitable participants for participatory sensing applications. In particular, reputation is used as a metric to determine how likely a user is to contribute data. Our work is different, in the sense that we exploit reputation as a means for evaluating the quality of data received from user devices.

The problem of verifying data received from user devices in participatory sensing has also been studied in [20, 21]. The focus of their work is to ensure that the data contributed by a mobile phone does indeed correspond to the actual data reported by the device sensors. In other words, they assume a threat model in which a malicious user or program may tamper with software running on the phones and corrupt the sensor data. Their solutions rely on an auxiliary trusted platform module (TPM), which vouches for the integrity of sensing devices. However, TPM-enabled mobile phones are yet to be mass produced and as such their solutions are not readily deployable. Moreover, these schemes do not necessarily overcome the particular situations targeted in this paper. For example, the TPM is unable to detect malicious behavior where the user may physically create interference that affects the sensor readings. Our work is thus complementary to the schemes proposed in [20, 21].

4. SYSTEM OVERVIEW

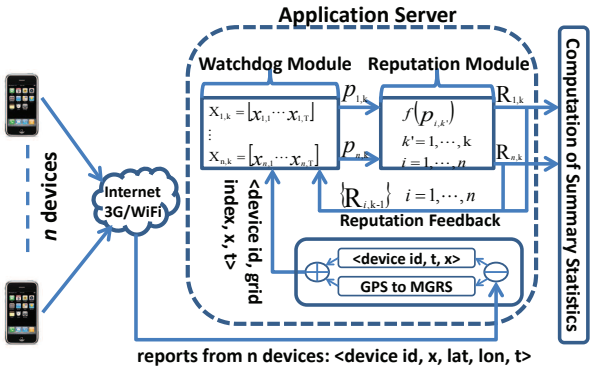

Figure 2: System architecture with information flow

In this section, we present an overview of the proposed reputation system in the context of a participatory sensing application. We provide detailed descriptions of the system components in Sections 5 and 6.

Fig. 2 presents a visual representation of our system architecture, which primarily consists of: (i) watchdog module and (ii) reputation module, both of which are implemented at the application server. Our system can readily work with any typical participatory sensing applications, which create summary statistics about the phenomenon being monitored (e.g., ambient noise as in the example in Section 2) from the sensor readings contributed by volunteer's mobile devices. In such an application, the central server exploits the inherent redundancy of samples, both in space and time. However, it is important to carefully define the granularity of space and time, over which multiple samples can be combined. For example, combining noise measurements taken 30 minutes apart from two closely located points is not meaningful. Neither is, combining noise samples measured at the same instant but from two distinct locations that are 100m apart. Hence, in our system, we assume that the spatial and temporal fields have been appropriately segmented into spatial *grids* and temporal *epochs*, such that only the sensor readings that belong to the same grid and epoch are aggregated by the server. The granularity of the grids and epochs are application-specific. In the rest of the paper, we present an application agnostic description of our system. We only consider scalar sensing modalities (e.g., noise) in this paper. We intend to investigate compatibility with vector sensor readings (e.g., images) in our future work.

We assume that the mobile phones of volunteers are instrumented with the appropriate program for collecting the readings of interest from the appropriate device sensor. Each sensor reading, which we simply denote by x, is tagged with the GPS coordinate (lat, lon) and system time (t) before being stored in the phone memory. The stored records of the form <device id, x, lat, lon, t>, are uploaded to the application server when the phone detects the presence of communication facilities, e.g., WiFi access point or 3G service. Upon receiving these samples, the application server first converts the GPS coordinates to the corresponding Military Grid Reference System (MGRS) grid index using the formulation specified in [22] and stores the reports (of the form <device id, grid index, x, t>) in a repository. The server then groups reports that belong to the same spatial grid (dimensions determined by the application) and forwards these to the watchdog module. In the rest of our description, we will only consider samples that belong to the same spatial grid. Hence, we neglect the grid index and simply refer to the sensor values as $x_{i,t}$, where i and t denote the device id and the time at which the sensor value is measured, respectively.

Let us assume that there are n devices contributing data within one particular spatial grid. The watchdog module processes sensor values from these n devices in epochs of duration T. More specifically, if we label each epoch as k, then sensor values from device i can be represented by a vector, $X_{i,k} = [x_{i,t}, ..., x_{i,t+T-1}]$, $\forall i$ with $t = (k-1) \times T + 1$, in that epoch [2]. The watchdog module executes an outlier detection algorithm on the vector $X_{i,k}$, and produces a set of *cooperative ratings*, $\{p_{i,k}\}$, for each device i in epoch k (the algorithmic details are described in Section 5). To build a long term perspective of the trustworthiness of each device, the cooperative ratings, $\{p_{i,k}\}$, act as inputs to the subsequent reputation module, wherein, they are further analyzed by a reputation function. For each epoch k, the reputation module incorporates past cooperative ratings (e.g., $\{p_{i,k'}, k' = 1, \cdots, k\}$) and computes a *reputation score*, $R_{i,k}$, for each device i (details of this operation are given in Section 6). The server can use the reputation scores to compute summary statistics. For example, the average sensor values in an MGRS grid can be computed as follows,

$$\bar{x_t} = \sum_{i=1}^{n} R_{i,k} \times x_{i,t},\ (k-1) \times T < t \leq k \times T \qquad (1)$$

where the sensor values are weighted in proportion to the reputation of each device in the time epoch over which the senor values are measured.

In the above description, we assumed that both cooperative rating and reputation score are attached to the device. However, if a device simultaneously contributes data from multiple on-board sensors, then the above ratings and scores can be maintained on a per-sensor basis. This is a simple modification that can be readily adopted in our system.

5. WATCHDOG MODULE

The watchdog module accepts vectors of sensor values, $\{X_{i,k}\}$, as input and computes the cooperative ratings, $\{p_{i,k}\}$, for each device i during each time epoch k. The cooperative rating, which is a number between the range (0,1), can be inferred as the level of confidence that can be associated with the readings contributed by a device. The watchdog module produces $\{p_{i,k}\}$ by executing an outlier detection algorithm [23, 24]. Outlier detection algorithms can be broadly classified as either model-based or consensus-based techniques [10]. A model-based approach requires *a prior* knowledge of the underlying physical process in which the application is interested, which can be difficult to acquire in our context. On the other hand, consensus-based techniques work on group consistency and use the deviations from a common consensus to identify outliers. Since the server groups reports from multiple devices in a grid, we choose a consensus-based technique for our watchdog module. In our system, we employ the algorithm presented in [25] for computing robust averages in traditional sensor networks. Robust average is a type of average value where the impact of malicious/faulty sensors are minimized through smaller weighting coefficients. In our context, the weighting coefficients correspond to the cooperative ratings $\{p_{i,k}\}$ while the robust average can be viewed as a summary statistic (we include this for comparison in the evaluations in Section 7). We provide a brief overview of the algorithm below and refer the reader to [25] for details.

Following from earlier notation, the instantaneous (at each t) average values for epoch k can be expressed as follows,

$$r_t = \sum_{i=1}^{n} p_{i,k} x_{i,t},\ (k-1) \times T < t \leq k \times T \qquad (2)$$

where $p_{i,k} \geq 0$ is the rating for device i in epoch k and applies to all $x_{i,t}$ in that epoch. Note that, a device is considered cooperative if its $p_{i,k} \geq 1/n$ [25]. It has been shown in [25] that Eq. 2 becomes the robust average if $\{p_{i,k}\}$ are calculated as Eq. 3 and the algorithm presented in Fig. 3 is

[2]For simplicity we assume that all n devices are continuously generating samples. This need not be the case as participatory sensing relies on voluntary contributions. Hence, users have complete freedom to contribute whenever they want.

executed.

$$p_{i,k} = \frac{\frac{1}{\frac{\sum_{t=1}^{T}(x_{i,t}-r_t)^2}{\sum_{i=1}^{n}\sum_{t=1}^{T}(x_{i,t}-r_t)^2}+\epsilon}}{\sum_{j=1}^{n}\frac{1}{\frac{\sum_{t=1}^{T}(x_{j,t}-r_t)^2}{\sum_{i=1}^{n}\sum_{t=1}^{T}(x_{i,t}-r_t)^2}}} + \epsilon \quad (3)$$

As seen from Fig. 3, the algorithm is iterative in nature; it computes r_t and $p_{i,k}$ in each iteration and continues iterating until it has achieved convergence. In our implementation, convergence is observed when $|p_{i,k}^{l} - p_{i,k}^{l-1}| < 0.0001$. Note that, we use this algorithm as an illustrative example but there are other consensus-based techniques that can be used instead, e.g., those discussed in [23, 24].

Let $p_{i,k}^{l}$ and r_t^{l} be the values of $p_{i,k}$ and r_t at the l^{th} iteration, respectively

1. Initialize $l = 0$ and $p_{i,k}^{l} = \frac{1}{n}$
2. Compute r_t^{l+1} from $p_{i,k}^{l}$ using Eq. 2
3. Compute $p_{i,k}^{l}$ from r_t^{l} using Eq. 3
4. $l \leftarrow l + 1$
5. Start from Step 2 if no convergence

Figure 3: Iterative outlier detection algorithm

6. REPUTATION MODULE

In Section 5, we presented an outlier detection algorithm that uses the samples in each epoch (i.e., the vector $X_{i,k}$, $1 \leq i \leq n$; $\forall k$) to produce the corresponding epoch-specific cooperative ratings, $\{p_{i,k}\}$, for devices in the same MGRS grid. In this section, we outline the design of our reputation module and show how it makes use of these epoch-based ratings to build a long term (i.e., over successive epochs) view of the trustworthiness of each device. In particular, we introduce the Gompertz function and demonstrate how it can be used to generate device reputation scores.

Prior to presenting the design of our reputation module, let us take a small detour and discuss how we use reputation in social situations. We tend to *gradually* build up trust in another person after several instances of trustworthy behavior. However, we *rapidly* tear down the reputation for this individual if we experience dishonest behavior on their part even in a handful of occasions. Since participatory applications are largely people-centric, it is logical to consider a similar approach for the evolution of reputation in such applications. We have selected to use the Gompertz function for computing reputation scores, since it is particularly well-suited to model this behavior.

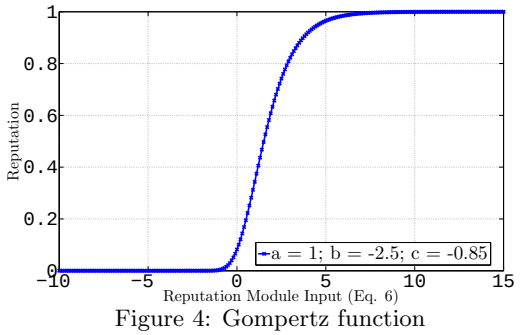

Figure 4: Gompertz function

The Gompertz function used in this work is plotted in Fig. 4 and is algebraically defined as follows:

$$R_{i,k}(p'_{i,k}) = ae^{be^{cp'_{i,k}}} \quad (4)$$

where a, b and c are function parameters. The parameter a specifies the upper asymptote, b controls the displacement along the x axis and c adjusts the growth rate of the function. The output of the function (and the reputation module), denoted by $R_{i,k}$, is a number in the range of 0 and 1 (inclusive) and represents the reputation score for device i in epoch k. The input of the Gompertz function (i.e., the right-hand side of Eq. 4), requires some elaboration. The input needs to reflect the fact that reputation is the result of aggregating historical device information (i.e., $p_{i,k'}$, $k' = 1 \cdots k$). Further, the aggregating process must account for the fact that the most recent information is more relevant than the past. Finally, note that $p_{i,k} \geq 0$ but the x-axis of Fig. 4 extends to negative numbers. Hence, $p_{i,k}$ needs to be mapped to the interval [-1,1]. In light of the above considerations, we first normalize the watchdog output (so that $-1 \leq p_{i,k} \leq 1$), as follows,

$$p_{i,k}^{norm} = \frac{2(p_{i,k} - min\{p_{i,k}\}_{i=1}^{n})}{max\{p_{i,k}\}_{i=1}^{n} - min\{p_{i,k}\}_{i=1}^{n}} - 1 \quad (5)$$

where $max\{p_{i,k}\}_{i=1}^{n}$ and $min\{p_{i,k}\}_{i=1}^{n}$ represent the maximum and minimum cooperative ratings from the watchdog module in epoch k, respectively. We can now express the input of the Gompertz function as follows:

$$p'_{i,k} = \sum_{k'=1}^{k} \lambda^{(k-k')} p_{i,k'}^{norm} \quad (6)$$

where the summation is used to facilitate the aggregation of historical information while the exponential term, $\lambda^{(k-k')}$ with $0 < \lambda \leq 1$, reduces the impact of past data (i.e. achieves ageing). In this sense, λ is equivalent to the *ageing weight* introduced in [10]. Hence, we follow the same nomenclature in the rest of this paper.

As mentioned earlier, it is desirable to have asymmetrical rates for improving and reducing reputation scores. This feature can be easily facilitated by our formulation of the function input. In particular, we implement this feature by replacing λ in Eq. 6 with two different ageing weights. The standard weight $\lambda_{standard}$ applies to cooperative devices (i.e., those with $p_{i,k} \geq 1/n$) while the penalty weight $\lambda_{penalty}$ applies otherwise (i.e., devices with $p_{i,k} < 1/n$). Note that, $\lambda_{penalty} > \lambda_{standard}$. The difference in ageing weights means that the summation term in Eq. 6 is dominated by negative $p_{i,k'}^{norm}$. Thus, it requires a device to act cooperatively (thus obtains positive $p_{i,k'}^{norm}$) in more occasions to neutralize its past non-cooperative behavior.

The reputation module computes device reputations using Eqs. 4-6 and provides the results to the application server. The server can use device reputation in several ways. For example, the server can compute the community average for each grid by weighting the samples from devices according to the corresponding device reputation scores, as in Eq. 1. An obvious question that arises is: why can't the cooperative ratings, $\{p_{i,k}\}$, computed by the watchdog module directly be used in place of $\{R_{i,k}\}$ in Eq. 1? After all, a low value of $p_{i,k}$ is a good indication that a particular device is not contributing good quality data and hence, its contributions should be given lower weights in the computation of community summary statistics. In what follows, we

sketch a high-level comparison between these two methods, which establishes the foundation for the evaluation results to be presented in Section 7.2. With cooperative ratings, the server's perception about devices are based on per-epoch approximations of their cooperativeness; it has no other information to validate whether the approximations have been accurate. On the other hand, reputation scores embody the server's view of the devices over several successive time epochs. Thus, the approximations made by the reputation module are not only more representative (from multiple observations) but also more objective (due to weighted averages).

7. EXPERIMENTAL EVALUATIONS

In this section, we detail the steps taken to evaluate the effectiveness of our reputation scheme. We describe the experiment setup in Section 7.1. In Section 7.2, we present results from a subset of our experiments that highlight the effectiveness of using Gompertz reputation. We also compare our results with those using Beta reputation.

7.1 Experimental Description

We evaluate our reputation system by incorporating it within a real-world participatory sensing application. As in the motivating example presented in Section 2, we consider a noise monitoring application, which relies on volunteers to contribute ambient noise level using their mobile phones. In the experiments, ambient noise is measured and recorded by a sound level meter program running on the mobile phones. The samples are sent via WiFi to a PC acting as the application server, which processes the reported measurements using the reputation system shown in Fig. 2. The system output (i.e., device reputation scores) is used by the server to compute the average noise level in the region of interest.

Recall that (see Section 4), our system assumes that space and time are segmented into application-specific grids and epochs, respectively. Since we use a noise monitoring application in our experiments, we follow the Australian acoustic standard [26] to determine the appropriate grid size. In accordance with the standard recommendations and the experiments conducted in [5], we choose to use a grid size of $30m \times 30m$. We assume a duration of 1 minute for the temporal epoch.

7.1.1 Hardware and Software

We used 8 Apple iPhones running OS version 3.1.3 in our experiments. We used an off-the-shelf application called SPL Graph [28], which enables the phone to function as a sound level meter (SLM) for collecting noise samples. This application samples audio signals from the built-in microphone at 48KHz and computes an A-weighted equivalent sound level (LA_{eq}) every second in dBA (this is consistent with the noise measurement guidelines in [26]). The readings are stored in a file and uploaded to the server via Wi-Fi. We also used an off-the-shelf commercial Centre 322 SLM [27] to measure the ground truth. This allows us to compare the output of our system with the actual sound level.

7.1.2 Procedure Overview

Recall that the goal of the reputation system is to evaluate the trustworthiness of the samples contributed by participating devices. As such, in our evaluations we artificially create situations where some of the devices contribute corrupted data, either inadvertently (reflecting careless behavior on part of the users or configuration errors) or maliciously. Due to space constraint, we present results from two experiments, with each experiment capturing a different usage scenario. In the first scenario, we do not consider malicious behaviors and only assume that a few devices contribute corrupted data inadvertently. The second experiment includes malicious users. We investigate if our reputation system is able to identify the devices contributing corrupted data (as is reflected by their reputation scores) and compute the community average in a robust manner.

The details of each scenario are explained in Section 7.2 where the corresponding results are discussed. Both experiments lasted for 60 minutes and were conducted in a grid of size $30m$ by $30m$ in the main library of UNSW. As a result, all 8 devices contribute samples that belong to the same spatial grid. The grid is approximately bound by a west-facing wall, north-facing windows and a set of newspaper shelves on the east. The 8 devices were randomly placed on any furnitures (3 study desks and a few bean bags) in the grid unless they were selectively positioned in the pockets (to simulate non-cooperative behavior). The Center 322 SLM was placed in the center of the grid to measure the ground truth in each experiment. Prior to each experiment, the clocks of all devices were synchronized with the 3G network. The microphones of all devices were calibrated according to the application guidelines. The SLM program on all devices were activated simultaneously at the start of each experiment and they continuously measured LA_{eq} every second until the end of the experiment.

The server processes the data contributed by the 8 devices in epochs of 60 seconds. More specifically, 60 noise samples recorded by each device within each epoch act as the input to the watchdog module. The watchdog module produces a cooperative rating ($p_{i,k}$) for each device using the algorithm presented in Fig. 3 and forwards the results to the reputation module. The reputation module computes the device reputation scores, $\{R_{i,k}\}$, by employing the Gompertz function as described in Section 6. The reputation scores are used by the application server to compute the average sound level in the grid using Eq. 1. We compare the performance of our scheme with the state-of-the-art Beta reputation [10]. For this, we compute the device reputation scores, $\{R_{i,k}\}$, using Beta distribution and use these scores to compute the average as per Eq. 1. We also include the robust average computed by the watchdog module in our comparison. Recall that, for computing the robust average, we replace $R_{i,k}$ by $p_{i,k}$ in Eq. 1. Finally, we also include the *raw* average in our comparisons, which is simply computed by averaging the samples from all devices without any associated weights.

We compare each of the above averages with the ground truth noise level recorded by the Center 322 SLM. We use the *Root Mean Square Error* (RMSE) to quantify the difference between the ground truth and each of the above averages. Given two vectors of average values (e.g., v_1 and v_2) the RMSE during the k-th epoch is defined as follows,

$$RMSE_k = \sqrt{\frac{\sum_{i=1}^{T}(v_{1,i} - v_{2,i})^2}{T}} \tag{7}$$

The mean RMSE for the duration of the entire experiment is obtained by averaging Eq. 7 over all epochs.

The following parameters are used in both experiments.

A single ageing weight, $\lambda = 0.7$, is applied to Beta reputation, while two different ageing weights, $\lambda_{standard} = 0.7$ (for cooperative contributions) and $\lambda_{penalty} = 0.8$ (for non-cooperative contributions) are used for Gompertz reputation. For both Beta and Gompertz reputations, an initial reputation of 0.5 is assigned to each device prior to the experiment. We assume the following parameters for the Gompertz function: $a = 1$, $b = -2.5$, and $c = -0.85$.

7.2 Evaluation Results

We now present the evaluation results from two of our experiments.

7.2.1 *Scenario One*

device 1	1	0	0	0	1	1
device 2	1	1	1	0	0	0
device 3	0	0	0	1	1	1
device 4	0	1	0	1	0	0
device 5	1	1	1	1	1	1
device 6	0	0	0	0	0	0
device 7	0	0	1	0	0	0
device 8	0	0	0	0	0	0

Table 1: Device placement matrix for the first scenario

We assume that the volunteers carrying the mobile phones can either place the phone on a piece of furniture (desk or bean bag) with the microphone exposed or in the user's pant pocket. The former represents the correct position for collecting sound samples, while the latter reflects a situation where the user has carelessly positioned the phone in such a way that the recorded samples will be corrupted (muffled by the pant fabric). In this scenario, we assume that each user makes a random decision about the placement of their phone every 10 minutes. We opt for a 10-minute interval because it gives us an opportunity to observe the evolution of the device reputation scores. Recall that we use an epoch of 1 minute, hence, reputations are computed every minute. Since the devices remain in the same position for at least 10 minutes, their reputation scores should evolve monotonically (i.e., continual increase or decrease). Table 1 denotes the placement configuration for all 8 phones in this scenario. Each column represents a 10-minute interval. "1" and "0" denote in-pocket and exposed positions, respectively. For example, device 3 is placed on the desk for the first 30 minutes and inside the pocket for the remaining 30 minutes.

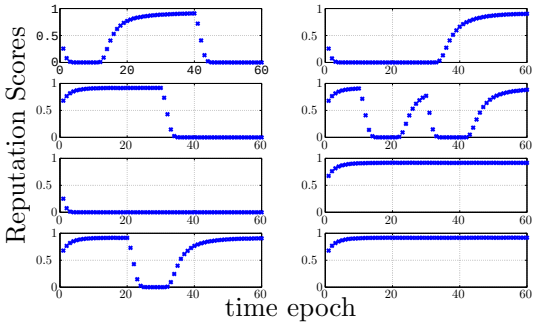

Figure 5: Evolution of Gompertz reputation for scenario 1. From top left to bottom right in left to right direction: device 1 to device 8.

Fig. 5 shows the evolution of reputation scores using the Gompertz function. Comparing Fig. 5 with Table 1, one can readily observe that the reputation scores perfectly track the device positions. We use device 1 as an illustrative example. In the experiment, this device was first placed in the pocket for 10 minutes, then it was on the table for the next 30 minutes and was finally moved to the pocket for the remaining 20 minutes. Its reputation score follows a similar pattern: it decreases in the first 10 minutes, gradually reaches the peak 40 minutes into the experiment and then declines until the end of experiment. We also observe that this device has been punished severely (reputation drops from 0.9 at $t = 40$ to 0 at $t = 44$), but rewarded gradually (it takes more than 20 minutes after $t = 10$ to improve reputation from 0 to 0.9) over the course of experiment. This behavior is a direct result of the different ageing weights used in our system as discussed in Section 6. It is worth reminding the reader that the server does not know the ground truth, i.e., it is not aware of the device positions. The reputation scores reflect the system's perception of the quality of data contributed by each device.

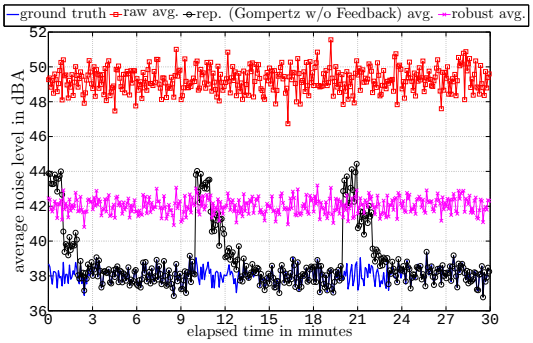

Figure 6: Average noise level for scenario 1

Fig. 6 compares the raw, robust and Gompertz averages with the ground truth for the first 30 minutes of the experiment (we omit the second half of the experiment since it shows similar results). As can be observed, the raw average is significantly different from the ground truth, since contributions (good and bad) from all devices are considered with equal importance. On the other hand, average computed using Gompertz reputation approximates the ground truth very closely (except for a few short periods, which we explain shortly). Finally, the robust average appears to deviate from the ground truth more significantly. We now proceed to explain this difference.

Devices	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
1 (CR)	0.045	0.045	0.044	0.045	0.044
2 (CR)	0.045	0.043	0.046	0.045	0.045
5 (CR)	0.043	0.045	0.044	0.043	0.045
1 (RS)	0.058	0.018	0.003	0.001	0.000
2 (RS)	0.058	0.017	0.004	0.001	0.000
5 (RS)	0.057	0.017	0.003	0.000	0.000

Table 2: Evolution of CR & normalized RS for devices 1, 2 and 5

Table 2 shows the cooperative ratings (CR) and the normalized Gompertz reputation scores (RS) assigned to devices 1, 2 and 5 in the initial 5 minutes of the experiment. Note that, these 3 devices were placed in the pocket for this interval (see Table 1) and as such, their samples should not be included in the calculation of average noise level. Recall that, the watchdog module computes CR based solely on the group consensus in one epoch, which means that the watchdog would produce similar CR values as long as the group conditions (i.e., number of devices and the placement

of these devices) remain constant. Since devices 1, 2 and 5 were retained in the pockets (and the other 5 devices remained exposed) during this interval, their CR values are almost identical, as shown in Table 2. Note that, the watchdog module identifies these devices as non-cooperative (since their CR values are $< 1/8$). However, the corresponding CR values allow their data to collectively account for about 13% of the robust average and consequently cause larger errors in estimating the ground truth. On the other hand, the reputation module assigns progressively smaller RS to these devices in the same period, which means their contributions to the Gompertz average diminish as time advances. In fact, at $t = 4$, these devices only account for $< 1\%$ of the Gompertz average and thus allow the server to better estimate the ground truth. This example serves as a strong justification for the use of a reputation system.

In Fig. 6, we observe that the Gompertz average deviates from the ground truth for a short duration beginning with $t = 11$ and $t = 21$, which correspond to the time instants just after the device positions are changed (see Table 1). This difference can be attributed to the fact that the reputation module requires some time to learn and adjust to possible transitions in the context of the devices. Consider device 4 as an example. This device was moved into the pocket at $t = 10$ and started contributing corrupted noise samples. As a result, its reputation begins to decrease. However, as seen from Fig. 5, it takes 3 minutes for the reputation module to learn about this and decrease the corresponding RS to zero.

Type of Average	Scenario 1	Scenario 2
Raw	11.28	8.23
Robust	4.02	4.29
Beta	2.33	4.27
Gompertz	0.73	3.73

Table 3: Mean RMSE for the entire experiment (in dBA)

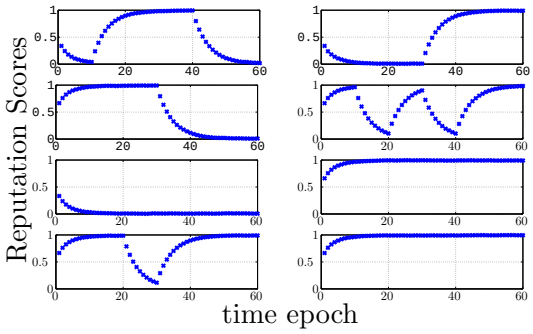

Figure 7: Evolution of Beta reputation for scenario 1. From top left to bottom right in left to right direction: device 1 to device 8

Table 3 summarizes the mean RMSE relative to the true values over the duration of the entire experiments. As is obvious, the raw average has a very large error. The Gompertz average reduces the estimation error by a factor of 5 and factor of 3 in comparison with the robust average and Beta average, respectively. To understand the performance difference between the Beta and Gompertz averages, we need to examine the RS values that are used by the server in the calculation of these two averages. Fig. 7 shows the evolution of Beta reputation scores for all devices. Comparing Fig. 7 with Fig. 5, we can see that Beta reputation takes a less aggressive approach in penalizing non-cooperative devices. As before, consider devices 1, 2 and 5. Observe that with Beta reputation, their RS values are non-zero at $t = 4$. This allows their contributions to still account for about 6% of the Beta average (cf. <1% in the case of Gompertz reputation), which results in the higher estimation error as seen in Table 3.

7.2.2 *Scenario Two*

In the previous scenario, we assumed that the only source of sensor data corruption was due to inadvertent actions on part of the users. In the second scenario, we extend the evaluations to include malicious activities. We use devices 7 and 8 as instances of adverse users and consider two types of malicious behavior. We assume that device 7 attempts to mislead the application server by artificially adding a constant offset of 30dBA to the values reported by the SLM, while device 8 synthetically adds random Gaussian noise to the SLM measurements[3]. The behavior of the other six non-malicious users are specified by a random matrix of dimension 6×60 (cf. 8×6 as shown in Table 1). As in the first scenario, the elements of the matrix are either "1" or "0" (picked randomly), which denote the in-pocket and exposed positions, respectively.

Table 3 summarizes the evaluation results. As can be observed, Gompertz reputation still generates the closest estimation to the ground truth in comparison with the robust and Beta averages. However, it is worth noting that the magnitude of estimation error has increased by 39% for the Gompertz average when compared with that in scenario 1. To explain this increase in magnitude, let us consider Fig. 8, which plots the evolution of reputation scores computed by the Gompertz function for devices 7 and 8. It is clear that, while our reputation system correctly identifies device 7 as a disreputable device, it fails to identify device 8 as a malicious one. As a result, the malicious data contributed by device 8 are included in the final calculation of Gompertz average, leading to an increase in the estimation error. We are currently exploring a device revocation scheme based on device reputation to address this problem. The rationale behind this approach is that, if a misbehaving device can be identified (e.g., via its associated reputation score) as it becomes malicious, the server should promptly remove its data from the reputation system (i.e., both watchdog and reputation modules) in the next time epoch. This prevents untrustworthy contributions from affecting the computation of common consensus in the watchdog module. The more accurate common consensus allows the server to compute more precise device cooperative ratings, $\{p_{i,k}\}$, which in turn result in the determination of more reliable device reputation scores, $\{R_{i,k}\}$.

We conclude by emphasizing that the choices of malicious behavior in this scenario are neither exhaustive, nor do we claim that our scheme can assure the application server of a similar level of performance in other types of ill-intended activities. In fact, we envision that Gompertz reputation will not function properly under more elaborated and planned attacks. For instance, if a malicious user is in control of more than half the devices in the same spatial grid, the watchdog module (see Section 5) will not produce meaningful cooperative ratings (since genuine devices become the

[3]We assume that the device users are able to modify the device software to launch these attacks, since the phone is not equipped with a TPM (see our discussion in Section 3)

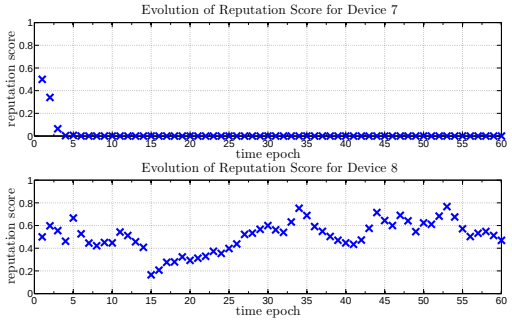

Figure 8: Evolution of Gompertz reputation for devices 7 & 8 in scenario 2

minority and are classified as non-cooperative). Gompertz reputation is also vulnerable to trial-and-error attacks where malicious user attempts to find the parameter values used by the application server. This would allow him to craft an attack where he precisely times the number of corrupted data contributions that he can relay to the server before his reputation is reduced to the extent that he becomes disreputable. However, since our solution punishes bad behavior more aggressively than Beta reputation, a malicious user has a much narrower window in which he can poison the final results if he succeeds to launch a trial-and-error attack.

8. CONCLUSIONS

In this paper, we made the case for evaluating device trustworthiness in participatory sensing applications and motivated the need for a reputation system. We proposed a reputation system that employs the Gompertz function for computing reputation scores. We experimentally evaluated our system by incorporating it within a real-world noise monitoring application. Experimental results demonstrate that our scheme achieves three-fold improvement in comparison with the state-of-the-art Beta reputation scheme.

9. REFERENCES

[1] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, M. B. Srivastava, "Participatory Sensing", in *Proceedings of the World Sensor Web Workshop, in conjunction with ACM SenSys'06*, November 2006.

[2] B. Hull, V. Bychkovsky, Y. Zhang, et. al., "CarTel: A Distributed Mobile Sensor Computing System", in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys'06)*, November 2006.

[3] E. Paulos, R. Honicky, E. Goodman, "Sensing atmosphere", in *Proceedings of the Workshop on Sensing on Everyday Mobile Phones in Support of Participatory Research, in conjunction with ACM SenSys'07*, November 2007.

[4] N. Maisonneuve, M. Stevens, M. E. Niessen, and L. Steels, "Noisetube: Measuring and mapping noise pollution with mobile phone", In *ITEE 2009 - Information technologies in Environmental Engineering*, Springer Berlin Heidelberg, May 2009. *Proceedings of the 4th International ICSC Symposium* in Thessaloniki, Greece, May 2009.

[5] R. Rana, C.T. Chou, S. Kanhere, N. Bulusu and W. Hu, "Ear-Phone: An End-to-End Participatory Urban Noise Mapping System", in *Proceedings of IPSN'10*, April 2010.

[6] S. Eisenman, E. Miluzzo, N. Lane, R. Peterson, G. Ahn and A. Campbell, "The Bikenet Mobile Sensing System for Cyclist Experience Mapping", in *Proceedings of SenSys'07*, November 2007.

[7] S. Reddy, A. Parker, J. Hyman, J. Burke, D. Estin and M. Hansen, "Image Browsing, Processing and Clustering for Participatory Sensing: Lessons from a DietSense Prototype" In *Proceedings of the Workshop on Embedded Networked Sensors (EmNetS)*, June 2007.

[8] Y. Dong, S. S. Kanhere, C. T. Chou and N. Bulusu, "Automatic Collection of Fuel Prices from a Network of Mobile Cameras", In *Proceedings of IEEE DCOSS 2008*, June 2008.

[9] S. Sehgal, S. S. Kanhere and C. T. Chou, "Mobishop: Using Mobile Phones for Sharing Consumer Pricing Information", Demo Paper in *Proceedings of IEEE DCOSS 2008*, June 2008.

[10] S. Ganeriwal and M. Srivastava, "Reputation-based Framework for High Integrity Sensor Networks", in *ACM Transactions on Sensor Networks (TOSN)*, Vol. 4, No. 3, May 2008

[11] P. Resnick and R. Zechhauser, "Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System", in *Working paper for the NBER workshop on empirical studies of electronic commerce.*

[12] A. Ghose, P. G. Ipeirotis, and A. Sundararajan, "Opinion Mining using Econometrics: A Case Study on Reputation Systems", in *Proceedings of the Association for Computational Linguistics (ACL)*, 2007.

[13] S. Buchegger and J. Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol", in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2002.

[14] P. Michiardi and R. Molva, "Core: a COllaborative REputation mechanism to enforce node cooperation in mobile ad-hoc networks", in *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, 2002.

[15] S. Buchegger and J. Y. Le Boudec, "Coping with False Accusations in Misbehavior Reputation System for Mobile Ad-hoc Networks", EPEL Technical Report Number IC/2003/31, 2003.

[16] S. Buchegger and J. Y. Le Boudec, "The Effect of Rumor Spreading in Reputation Systems for Mobile Ad-hoc Networks. in *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks"*, March 2003.

[17] A. Jøsang and R. Ismail, "The Beta Reputation System", in *Proceedings of the 15th Bled Electronic Commerce Conference*, June 2002.

[18] A. Gelman, J. B. Carlin, H. S. Stern and D. B. Rubin. *Bayesian Data Analysis.* 2003, Champman and Hall.

[19] S. Reddy, D. Estrin, M. Srivastava, "Recruitment Framework for Participatory Sensing Data Collections", in *Pervasive'10*, May 2010.

[20] A. Dua, N. Bulusu, W. Feng, W. Hu, "Towards Trustworthy Participatory Sensing", in *Proceedings of 4th USENIX Workshop on Hot Topics in Security (HotSec '09)*, August 2009.

[21] S. Saroiu and A. Wolman, "I Am a Sensor, and I Approve This Message", in *Proceedings of the Eleventh Workshop on Mobile Computing Systems and Applications, HotMobile'10*, February 2010.

[22] National Geospatial-Intelligence Agency (NGA). Datums, Ellipsoids, Grids, and Grid Reference Systems. *DMA TECHNICAL MANUAL*

[23] M. M. Breunig, H.-P. Kriegel, R.T. Ng, and J. Sander, "LOF: Identifying Density-based Local Outliers", in *Proceedings of the ACM SIGMOD Conference*, 2000.

[24] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, C. Faloutsos, "LOCI: Fast Outlier Detection Using the Local Correlation Integral", in *Proceedings of IEEE International Conference on Data Engineering*, March 2003.

[25] C. T. Chou, A. Ignjatovic, W. Hu, "Efficient Computation of Robust Average in Wireless Sensor Networks using Compressive Sensing", Technical Report: UNSW-CSE-TR-0915. `ftp://ftp.cse.unsw.edu.au/pub/doc/papers/UNSW/0915.pdf`

[26] Australia/New Zealand Standards Committee av/5. Australian Standard: Acoustics Description and Measurement of Environmental Noise. AS 1055.3 1997, Part 3 - Acquisition of Data Pertinent to Land Use.

[27] Center Technology Corp. Center c322. `http://www.centertek.com`

[28] SPL Graph. An Audio Level Chart Recorder for the iPhone and iPod Touch, `http://www.studiosixdigital.com/leq_graph.html`